New York University Tandon School of Engineering
Prof. Gustavo Sandoval

Intro to ML: Written Homework 2.

Release Date: 2/24/2024
Due Date: Sunday 3/9/2024 11:59pm

Discussion with other students is allowed for this problem set, but solutions must be written-up individually.

Problem 1: Regularization (12pts)

In this problem, we will look at how to exhaustively search over all possible subsets of features. You are given three python functions:

```
model = LinearRegression()  # Create a linear regression model object
model.fit(X,y)              # Fits the model
yhat = model.predict(X)     # Predicts targets given features
```

Given training data `Xtr,ytr` and test data `Xts,yts`, write a few lines of python code to:

(a) Find the best model using only one feature of the data (i.e. one column of `Xtr` and `Xts`).

(b) Find the best model using only two features of the data (i.e. two columns of `Xtr` and `Xts`).

(c) Suppose we wish to find the best k of p features via exhaustive searching over all possible subsets of features. How many times would you need to call the `fit` function? What if $k = 10$ and $p = 1000$?

Problem 2: Impacts of Regularization (12pts)

Consider the ridge regularized least squares regression problem $\|\mathbf{X}\boldsymbol{\beta} - \mathbf{y}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_2^2$ with different positive values of λ. Let $\boldsymbol{\beta}_1^* = \arg\min \|\mathbf{X}\boldsymbol{\beta} - \mathbf{y}\|_2^2 + \lambda_1\|\boldsymbol{\beta}\|_2^2$ and $\boldsymbol{\beta}_2^* = \arg\min \|\mathbf{X}\boldsymbol{\beta} - \mathbf{y}\|_2^2 + \lambda_2\|\boldsymbol{\beta}\|_2^2$.

(a) Prove that if $\lambda_1 \geq \lambda_2$ then $\|\boldsymbol{\beta}_1^*\|_2^2 \leq \|\boldsymbol{\beta}_2^*\|_2^2$. In words, increasing the regularization parameter *always* decreases the norm of the optimal parameter vector.

(b) Prove that if $\lambda_1 \geq \lambda_2$ then $\|\mathbf{X}\boldsymbol{\beta}_1^* - \mathbf{y}\|_2^2 \geq \|\mathbf{X}\boldsymbol{\beta}_2^* - \mathbf{y}\|_2^2$. In words, increasing the regularization parameter *always* leads to higher train loss, even if it might improve test loss.

(c) Suppose instead that we used LASSO regularization, so that Let $\boldsymbol{\beta}_1^* = \arg\min \|\mathbf{X}\boldsymbol{\beta} - \mathbf{y}\|_2^2 + \lambda_1\|\boldsymbol{\beta}\|_1$ and $\boldsymbol{\beta}_2^* = \arg\min \|\mathbf{X}\boldsymbol{\beta} - \mathbf{y}\|_2^2 + \lambda_2\|\boldsymbol{\beta}\|_1$. Do the above conclusions change when $\lambda_1 \geq \lambda_2$?

Problem 3: Naive Bayes (12pts)

Email	**Class**
”send us your password”	spam
”send us your review”	ham
”review your password”	ham
”review us ”	spam
”send your password”	spam
”send us your account”	spam

In class we worked with the dataset above. There are two versions of Naive Bayes for text categorization problems. One is **Bernoulli Naive Bayes** and the other is called **Multinomial Naive Bayes**. Bernoulli Naive Bayes uses the standard Naïve Bayes approach to classification, with binary-valued attributes, and is the one we saw in class. In Bernoulli, Naive Bayes, each word in the vocabulary is a binary-valued attribute. The value of the attribute is 1 if the word occurs in the document, and 0 otherwise.

Multinomial Naive Bayes uses the standard Naïve Bayes approach to classification, with integer-valued attributes. In Multinomial Naïve Bayes, we assume that for each category, there is a distribution over the words in the vocabulary. We view the generation of a document of length m in category C as the result of m independent choices from the distribution of words in category C.

In both versions of Naive Bayes, we can use the Laplace smoothing technique to avoid zero probabilities. The Laplace smoothing technique adds a small positive number to each count in the table. You can use any small positive number you want, but in our case we will use 1.

So for example in multinomial Naive Bayes we ask for the probability of the word "send" in the class "spam". We look at the table above and see that the word "send" appears 3 times in the class "spam" and 2 times in the class "ham". The spam class has 13 words in total and the size of the vocabulary is 6. The non smoothed 3/13 probability is 0.23076923076923078. The smoothed probability is (3+1)/(13+6) = 0.2857142857142857.

Now compute the following:

(a) What is your estimate of the probability that a randomly chosen email is spam?

(b) What is your estimate of P("password" | spam) using **Bernoulli Naive Bayes** with no smoothing?

(c) If we categorize using **Bernoulli Naive Bayes** with Laplace Smoothing, what is your estimate of the probability that the email "send us your account" is spam?

(d) If we categorize using **Multinomial Naive Bayes** with smoothing, what is your estimate of the probability that the email "review your account" is spam?

Problem 4: Logistic Regression (12pts)

A data scientist is hired by a political candidate to predict who will donate money. The data scientist decides to use two predictors for each possible donor:

- x_1 = the income of the person (in thousands of dollars), and
- x_2 = the number of websites with similar political views as the candidate the person follow on Facebook.

To train the model, the scientist tries to solicit donations from a randomly selected subset of people and records who donates or not. She obtains the following data:

Income (thousands \$), x_{i1}	30	50	70	80	100
Num websites, x_{i2}	0	1	1	2	1
Donate (1=yes or 0=no), y_i	0	1	0	1	1

(a) Draw a scatter plot of the data labeling the two classes with different markers.

(b) Find a linear classifier that makes at most one error on the training data. The classifier should be of the form,

$$\hat{y}_i = \begin{cases} 1 & \text{if } z_i > 0 \\ 0 & \text{if } z_i < 0, \end{cases} \qquad z_i = \mathbf{w}^\mathsf{T}\mathbf{x}_i + b.$$

What is the weight vector $\mathbf{w}$ and bias b in your classifier?

(c) Now consider a logistic model of the form,

$$P(y_i = 1|\mathbf{x}_i) = \frac{1}{1+e^{-z_i}}, \qquad z_i = \mathbf{w}^\mathsf{T}\mathbf{x}_i + b.$$

Using $\mathbf{w}$ and b from the previous part, which sample i is the *least* likely (i.e. $P(y_i|\mathbf{x}_i)$ is the smallest). If you do the calculations correctly, you should not need a calculator.

(d) Now consider a new set of parameters

$$\mathbf{w}' = \alpha \mathbf{w}, \quad b' = \alpha b,$$

where $\alpha > 0$ is a positive scalar. Would using the new parameters change the values $\hat{y}$ in part (b)? Would they change the likelihoods $P(y_i|\mathbf{x}_i)$ in part (c)? If they do not change, state why. If they do change, qualitatively describe the change as a function of α.

Problem 5: Tree Based Methods (12pts)

Consider a regression problem with training data $D = \{(X_i, y_i) : i = 1, \ldots, n\}$ where each X_i is a d-dimensional predictor vector and y_i is a continuous response. Answer the following:

(a) Describe the recursive binary splitting algorithm used to build regression trees. In your answer, explain how the algorithm chooses a split at a given node and derive the formula for the reduction in the Residual Sum of Squares (RSS) when splitting the data into two regions, R_l and R_r, based on a candidate rule $(X_j < s)$.

(b) Discuss the rationale behind cost complexity (weakest link) pruning. Define the cost complexity criterion used in pruning a large tree and explain how cross-validation is employed to select the optimal subtree.

(c) Compare a single decision tree with ensemble methods—specifically bagging, random forests, and boosting. In your discussion, address:

i. How ensemble methods reduce the variance (and sometimes bias) compared to a single tree.

ii. The trade-offs involved, particularly regarding interpretability versus prediction accuracy.

Problem 6: Applications (12pts)

Download the following from Google Drive. It contains:

1. A Jupyter notebook that you can run on your own computer or on the Colab environment. The notebook has further instructions and a template for you to fill in.

2. A subdirectory called `data` that contains the two data files

3. If you use Google Colab, then you can make a copy of the notebook here, however you will need to upload the `data` subdirectory to your Google Drive. Then, you will need to mount your Google Drive to the Colab environment. The following code will do this, but you will need to deal with paths appropriately.

```
        from google.colab import drive
        import pandas as pd

        drive.mount('/content/gdrive')
        path_to_insults = '/content/drive/MyDrive/data/'
        data = pd.read_csv(path_to_insults + 'train-utf8.csv')
        data.head(2)
```